Which Model for Which Role?

Cost-Efficient Pareto Recommendation for LLM Agentic Workflows

Qian Xie[1,2,*] Wenyue Hua[3,*] Yueli He[4] Sripad Karne[1] Armaan Agrawal[1] Nikos Pagonas[1] Eugene Wu[1] Kostis Kaffes[1] Tianyi Peng[1] Nairen Cao[5,4]

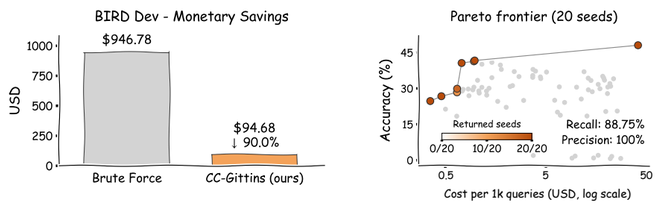
[1]Columbia University [2]Cornell University [3]Microsoft [4]New York University [5]Shanghai University of Finance and Economics [*]Equal contribution

Paper GitHub

Find the right LLM models for your AI agents. Explore quality–cost trade-offs with cost-efficient search.

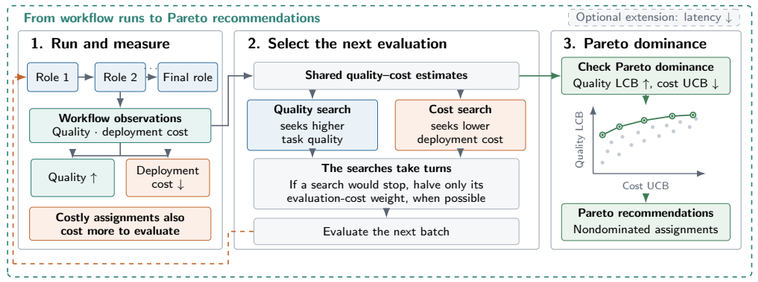
1 Find quality–cost trade-offs efficiently

For a four-role text-to-SQL workflow on BIRD Dev, we compare 75 model assignments across 1,534 queries.



CC-Gittins recovers nearly 90% of the Pareto frontier at 10% of the brute-force budget.

2 AgentOpt: evaluate, search, recommend



CC-Gittins is AgentOpt's core search method.

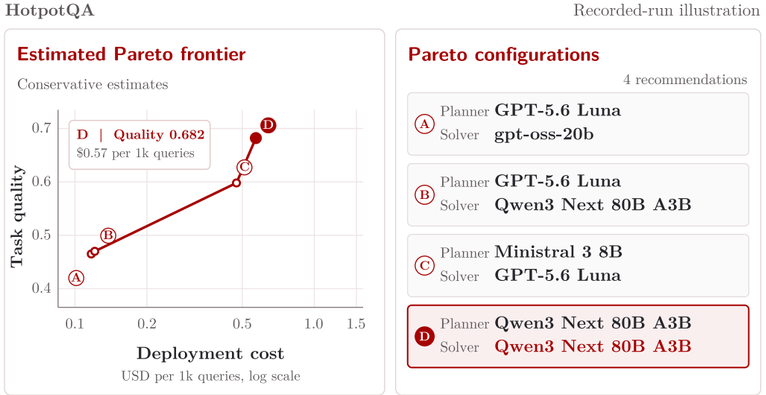
8 instances 2–5 roles up to 1,681 assignments

Data: HotpotQA and MathQA evaluated with AgentOpt + 6 from SCOPE.

Offline replay; 20 seeds; curves: mean ±2 SE.

3 AgentOpt user interface

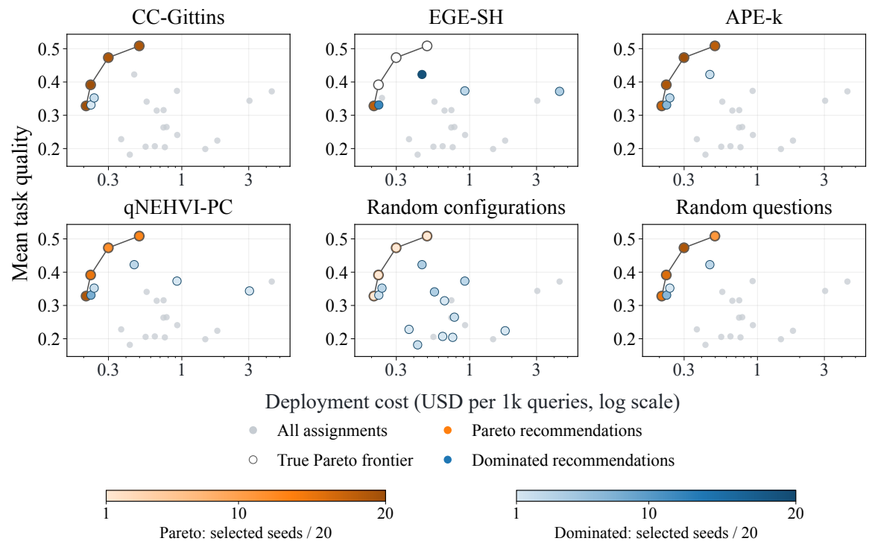
HotpotQA — Recorded-run illustration



12% of brute-force runs | **$1.9242** evaluation spend | **CC-Gittins**

4 Two objectives: quality + deployment cost

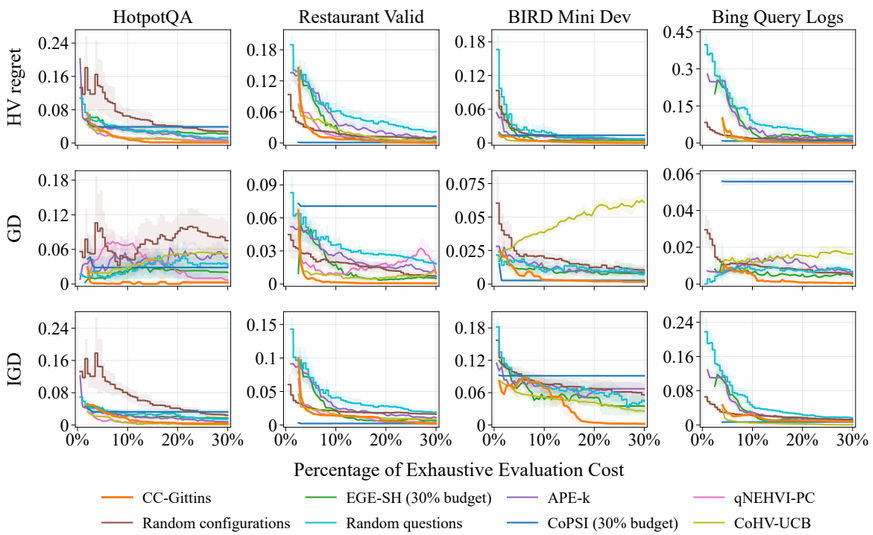
Stack Overflow, 10% evaluation cost: **CC-Gittins: 97% precision; 98% recall.**



Darker points: recommended in more seeds. Precision/recall: seed averages.

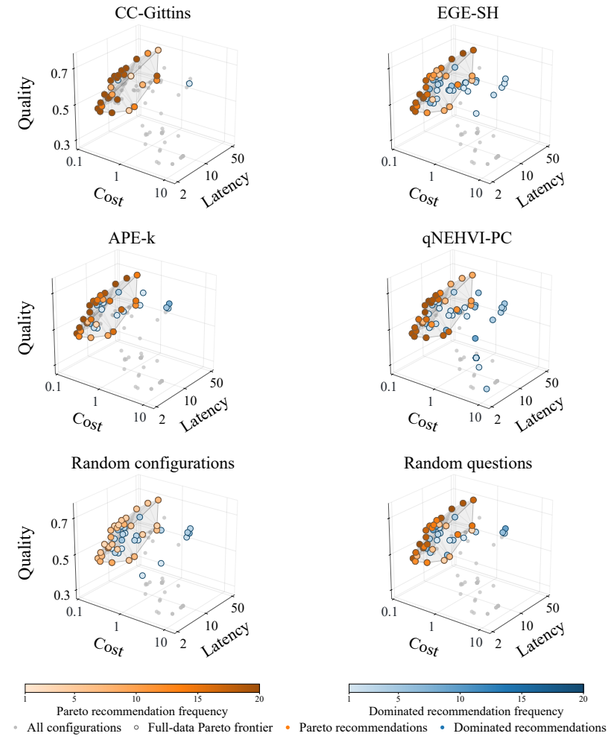
Quality–cost across four workflow families (2–5 roles)

All three metrics: lower is better.
HV regret: lost dominated volume.
GD: mean distance from recommendations to the true frontier.
IGD: mean distance from the true frontier to recommendations.



5 Three objectives: add latency

HotpotQA, 30% evaluation cost.



CC-Gittins: 99.6% precision; 80.3% recall.

Cost: USD per 1k queries; latency: LLM seconds/query (log scales).

Quality–cost–latency across evaluation budgets

